

**TITLE: IMPROVED METHOD FOR TRACKING AUDIT FILES
 SPANNING MULTIPLE TAPE VOLUMES**

FIELD OF THE INVENTION:

 The present invention generally relates to the
field of magnetic tape drives and in particular to an
improved method for tracking audit files spanning
5 multiple tape volumes.

BACKGROUND OF THE INVENTION:

A namespace is a set of names in which all names are unique. Namespace management is a well-defined method of inserting and deleting names into the namespace (with no duplicate names) and an unambiguous method of tracking names in that namespace. In the computing world, namespace management schemes can be applied to managing multiple files, each with unique filenames.

For disk storage devices, the rules for managing the namespace are clear. The disk namespace is called a disk directory and the elements of a disk namespace are filenames and all of them have to be unique. The operating system provides efficient mechanisms for inserting/deleting files into the disk namespace and for filename searches. The second (and a very important) characteristic of disks is that the namespace (i.e. directory) is maintained on the same storage medium as the elements of the namespace (i.e. filenames). The other characteristic of disk storage media is that they are not generally removable from a system and hence each individual disk device has its own unique name in a system.

Tape devices, on the other hand, differ significantly in terms of the characteristics mentioned above for disk devices. Tapes do not have a well-defined system providing a namespace management scheme. Users have to manage the namespace themselves. There are cataloging systems that do provide some form of management but they are limited when compared to disk namespace management systems. Moreover, tapes are removable storage media and hence the possibility of

having duplicate copies of the same tapes is a distinct reality.

Tape devices prove to be extremely useful when dealing with audit files (also known as transaction logs) and database recovery. Audit files contain a history of the changes made to an audited database and are a necessary component of the database recovery process. If audit files are not available when a database failure occurs, all changes made to a database since the last backup dump was taken might be lost. Because audit files can contain information for millions of transactions, audit files tend to be large. It is common to store backup audit files on tapedrives rather than waste hard disk on the database system server.

Because of the important role tape devices play in database recovery, and the deficiencies of managing namespaces when dealing with tape devices, the need arises for maintaining and tracking files spanning multiple reels of tape.

The method of the present invention addresses this need by introducing the concept of a Tapeset, or a grouping of audit files on tape. The biggest advantage that can be derived from the method of the present invention is when new audit files are being appended to an existing Tapeset or when audit files are being retrieved. Tapes using the method of the present invention are created under the constraint that all audit files on the tape (which may span multiple reels) must be stored sequentially by audit file number. Consequently every command to append audit files to these tapes needs to be verified for the aforementioned constraint. This is done by verifying whether the previous audit file (by
awk/appl/558L.doc

audit file number) for the database exists on some tape. If the constraint is verified, the method of the present invention then positions the tape just beyond the end of that file to prepare for appending the new audit file.

5 In the prior art, this was done by simply calling an operating system function to open the previous audit file. In other words, if audit file #5 was being appended then the prior art utility invoked an operating system function to open audit file #4. If audit file #4
10 is found, another operating system function (close with retention) is invoked to position the tape just beyond the end of the file.

Now, in today's data center environments with the multitude of tape drive hardware and the possibility
15 of multiple audit tapes being mounted on that hardware, it took an inordinate amount of time for the operating system to open the requested audit file. This is because the operating system sequentially scanned thru all the mounted audit tapes of the database for the requested
20 file. In many cases, several tape volumes have to be searched sequentially before the right tape volume with the audit file in question is found (Problem #1). This problem is extremely aggravated by the high capacity of today's tapes. Sometimes, scanning thru an entire reel
25 of tape takes more than an hour just to determine whether a particular file is present on the tape (Problem #2). In addition, if the audit file whose presence was being verified (in our case: audit file #4) were huge and worse, if it spanned multiple reels, it would involve
30 another large chunk of elapsed time to be taken for positioning the tape drive at the end of the file (Problem #3).

In the method of the present invention, the concept of the Tapeset is invoked again. If the "Append Command" is issued by the operating system (DBMS software), the current Tapeset number stored in the system control file for the database is used. If the
5 Append Command is issued by the operator, the Tapeset number has to be supplied.

One prior art method to which the method of the present invention generally relates is described in U.S.
10 Patent No. 5,982,572, entitled METHOD AND APPARATUS FOR ENABLING FAST ACCESS TO A LOGICAL BLOCK ON A TAPE MEDIUM. The prior art reference discloses a method and apparatus for fast access to any logical block on a media not containing logical block addressing. Categorizing marks
15 such as filemarks or setmarks are provided on the tape medium at various points along the medium. The medium is divided up into a plurality of physical blocks. To permit fast access to a logical block on the tape, a connection table in the form of a block map is provided
20 which establishes a relationship between logical blocks and the tapemarks, and defines physical positions of at least some of the tapemarks.

The method of the present invention differs from the above-mentioned prior art method because by
25 establishing unique names for sets of tape volumes using a mechanism called Tapeset; the Tapeset assigns a number that will indicate the first file in each tape volume; a MAXFILES PERTAPE value is then associated with each file so that locating a required file can be done more
30 efficiently. The prior art method, on the other hand, sets up tapemarks along the tape medium and divides the tape into several partitions without identifying the

files stored in these blocks. In order to have fast access using the prior art method, a connection table is provided which establishes the relationship between the partitioned blocks and tapemarks.

5 Another prior art method to which the method of the present invention generally relates is detailed in U.S. Patent No. 5,287,232, entitled AUTOMATIC TAPE SEARCHING METHOD. This prior art reference is an automatic (auto) tape searching method that searches the
10 required tape portion fast and automatically by repeating a sequence of operational procedures. The auto tape searching method comprises: a first stage which initializes the port of a microcomputer and the content of a RAM (random access memory), sets the initial values,
15 and controls interrupts at a power-on reset of a video tape recording system; a second stage which checks key input and, for auto random search (ARS) key, performs the auto random search function; a third stage which performs mode checking and, if a present mode needs to accompany
20 the mechanism operation, controls the present mode; and a fourth stage which switches the mode and checks the sensor and an emergency state to put the system in stable condition.

 This prior art method is a hardware solution to
25 the quick search problem. Its proposed solution is to automate a searching algorithm that will search the required tape portion; this prior art method utilizes four stages that manipulates RAM content and key inputs, performs mode checking, and evaluates system state
30 conditions. The method of the present invention, on the other hand, is software based but achieves the same goal

utilizing a nomenclature system that efficiently locates a specific data file.

Still another prior art clustering system to which the method of the present invention generally relates is detailed in U.S. Patent No. 5,757,571, entitled FLEXIBLE-CAPACITY SCALING FOR EFFICIENT ACCESS OF ORDERED DATA STORED ON MAGNETIC TAPE MEDIA. This prior art method discloses various data storage formats that help to efficiently locate, read, and write user data stored on magnetic tape media. A tape is formatted by writing multiple segment-headers, free from any interleaved access of user data. Adjacent segment-headers are spaced by a predetermined interval to define multiple data storage segments. Segment-headers all contain a unique key, which is copied into a key index to identify valid segments. After formatting, normal tape accesses can be performed. Without erasing any old headers or data, a new formatting scheme can be established by writing new segment-headers on the tape. The new segment-headers include a new unique key, replacing the previous key in the key index. Previous segment-headers stored on the tape are ignored, since they lack the updated key. Segments may be selectively grouped to provide independently addressable partitions. Mapping between segments and partitions can use a fixed relationship (e.g. one-to-one), or each partition may be variably sized according to the amount of data to be stored therein. Variable-sized partitions may be automatically padded with a selected number of empty segments. Another feature is flexible-capacity scaling, which distributes an ordered set of device blocks on a multi-track magnetic tape medium. The device blocks are

awk/appl/558L.doc

bi-directionally stored in a continuous configuration of multiple adjacent stacked serpentine patterns occupying some or all of the tape. This configuration permits sequential access of all device blocks without advancing
5 the tape medium to skip over regions between adjacent device blocks.

The key difference between the prior art method and the method of the present invention is the organization of file-positioning information provided by
10 the method of the present invention. The method of the present invention uses a disk directory file to access files, identify files, and resolve name conflicts when multiple tapes are created with the same name. The disk directory file contains the name of the first file in
15 each volume; a serial number is also associated with each volume. Together these two identifiers are used to limit the search through tape drives.

Still another prior art clustering system to which the method of the present invention generally
20 relates is detailed in U.S. Patent No. 5,572,378, entitled DIRECT FILE ACCESS SYSTEM FOR MAGNETIC TAPE. This prior art method discloses a direct file access system for a magnetic tape where all data files begin at a designated location on the tape. The direct file
25 access system may be used with a reduced rewind data configuration to decrease data access time. The reduced rewind data configuration divides data files into generally equal portions so that data files begin and end at a designated location on the tape, eliminating rewind
30 sequences. A method and system for reducing the number of tape retensioning passes is included to further decrease access time.

This prior art method divides data into equal portions so that all data files start and end on designated locations on the tape. A reduced rewind scheme is then applied to zip through these data blocks to find data. This method, however, uses a linear search that must run through an entire block to locate a specific data file. The method of the present invention, however, assigns numeric values to each file in a data block, reducing the search time from $O(n)$ to $O(1)$. $O(n)$ is a notation for specifying that the performance of an algorithm depends on the length or number of inputs "n". $O(1)$ signifies that regardless of "n" the algorithm performs in time similar to when $n=1$.

Yet another prior art clustering system to which the method of the present invention generally relates is detailed in U.S. Patent No. 6,081,875, entitled APPARATUS AND METHOD FOR BACKUP OF A DISK STORAGE SYSTEM. This prior art reference is a backup system and method providing for the creation of a reconciled snapshot backup image of a database while the database, residing on a disk array system, is in use by users. A backup computer running a commercial backup utility is connected between the array system and a tape storage system. While the backup is underway, write requests to the database are suspended until the data currently in those data blocks is copied and stored in an original data cache. Here, the disk system address of the copied block and a pointer to the location of the block in the cache are stored in a map. The backup utility incrementally reads portions of the database from the disk system and forwards those portions to the tape system. Prior to each portion being forwarded to the tape system, all data blocks in the portion which have an

address that corresponds to the address of a block in the cache are discarded and replaced with the data from the cache for that address.

5 This prior art reference is a method to backup data on a disk storage system. It does not deal with the problem addressed by the method of the present invention, that is to say, providing for the efficient locating of specific data files within a storage system.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215

SUMMARY OF THE INVENTION:

It is therefore an object of the present invention to provide a method for tracking files spanning multiple tape reels using fast-locate capable tapedrives.

5 Still another object of the present invention is to expedite the retrieval of audit files contained on tape when performing a database recovery.

Still another object of the present invention is to create a common name for grouping of audit files spanning one or more tape volumes.

Yet another object of the present invention is to provide positioning information of audit files contained on tape.

The method of the present invention is useful in a computer system including a server accessing a database and connected to a magnetic tape drive. An associated program (the COPYAUDIT utility) executes a method for locating audit files of a database that are backed-up on tape. The method includes the steps of creating a Tapeset for the group of audit files, then initializing a disk directory file to hold positional information of the Tapeset, and finally, locating each of the audit files within the group of audit files using the positional information contained in the disk directory file.

BRIEF DESCRIPTION OF THE DRAWINGS:

FIG. 1 is a generalized block diagram of a system that may use the method of the present invention;

FIG. 2 is a generalized block diagram illustrating the transfer of audit files from disk to tape;

FIG. 3 is a block diagram illustrating a sample of Tapesets and their respective disk directory file;

FIG. 4 is a flowchart that illustrates the steps for creating an initial Tapeset;

FIGS. 5A and 5B are combined to form a flowchart that illustrates the steps for appending an audit file to a Tapeset;

FIG. 6 is a flowchart that illustrates the steps for updating the disk directory file;

FIGS. 7A and 7B are combined to form a flowchart that illustrates the steps for retrieving an audit file from a Tapeset.

DESCRIPTION OF PREFERRED EMBODIMENT:

Before proceeding with a detailed description of the method of the present invention, a background discussion of Tapesets and disk directory files may be helpful. The concept of a Tapeset was specifically created for the present invention. A Tapeset is defined as a grouping of files on a tape. A Tapeset can span multiple reels. Files within a Tapeset can start at the end of one tape reel, and continue at the beginning of another tape reel. The Tapeset number is the identification of tape volumes on which several audit files reside.

A disk directory file is created on the mainframe server to correspond with each Tapeset. The disk directory file will have the Tapeset number as one of the nodes. Thus, there is a mapping between the disk file directory and the Tapeset name. Since the real problem is to get to the right directory in the first place, the Tapeset number is a convenient way to access the right directory. The disk directory also contains positioning information for each file within the Tapeset. This positioning information is used as parameters to fast-locate capable tape drive systems that function to provide fast positioning within a tape volume.

Referring now to the drawings and FIG. 1 in particular, a block diagram of a computer system is shown including server 15 typically running DMSII Software 16 and COPYAUDIT Software 17. DMSII represents a Data Management System developed by Unisys Corporation described in a publication entitled Unisys e@action Enterprise Database Server Extended Edition for Clearpath

MCP, and published November, 2000. The COPYAUDIT Software 17 contains the software logic involved for the present invention. Server 15 is connected to database 14. Database 14 generates a collection of audit files 10.

5 Audit files 10 maintain an event history of all transactions occurring on server 15. In this example, audit files 10 are comprised of AuditFile1 11, AuditFile2, 12, and AuditFile3, 13. Server 15 is also connected to a magnetic tapedrive 18, which is used to

10 store a backup copy of the audit files 10. Magnetic tape drive 18 contains a FAST-LOCATE capability, which enables fast cueing to a certain position within a particular tape volume. The FAST-LOCATE capability is a combination of the system software interfaces and tape drive

15 capability. The system software interfaces are described in Master Control Program (MCP) System Interfaces Programming Reference Manual, published in October 1999, by the Unisys Corporation.

With reference to FIG. 2, a diagram

20 illustrating the transfer of audit files from disk to tape is shown. The audit files on disk 10 are sent through COPYAUDIT Software 17 residing on server 15, and are backed up as audit files on tape at step block 22 using magnetic tape drive 18. COPYAUDIT is a program

25 developed by Unisys Corporation, Blue Bell, Pennsylvania, and described in a publication entitled "Enterprise Database Server for ClearPath MCP Utility Operations Guide", published in November, 2000.

Referring now to FIG. 3, a block diagram

30 illustrating a sample of Tapesets and their respective disk directory file is shown. The first Tapeset 30

contains three tape volumes of audit files. The first volume 34 contains audit files sequentially numbered 1 through 6. The second volume 35 contains audit files numbered 7 through 10A. The third volume 36 contains audit files 10B through 15. The second volume 35 has the beginning portion of audit file 10 (10A), while the third volume 36 contains the ending portion of audit file 10 (10B). The first Tapeset 30 has a disk directory file 32 associated with it that keeps track of tape volume and audit file position information.

With reference to FIG. 3, a second Tapeset 31 contains three tape volumes of audit files. The first volume 37 contains audit files numbered 16 through 20A. The second tape volume 38 contains audit files numbered 20B through 25. The third tape volume 39 contains audit files numbered 26 through 30. The second Tapeset 31 has a disk directory file 33 associated with it that keeps track of tape volume and audit file position information.

With reference to FIG. 4, a flowchart that illustrates the steps for creating an initial Tapeset is shown. During this process, a tape volume is created, a disk directory initialized, and an initial collection of audit files are written to tape. The process begins with start bubble 40 and continues with a process step (block 41) of creating a tape volume marker file. Each physical reel (also known as a volume) of a Tapeset contains a tape volume marker file. The process continues by initializing a disk directory file (block 42). In initializing the disk directory file, the process first creates a disk directory file, then inserts a directory record as the first record in the disk directory file. Next, the process continues with an inquiry as to whether

or not another audit file from audit files 10 (FIG. 1) on database 14 needs to be appended to the list of audit files on tape step block 22, FIG. 2 (diamond 43). If the answer to this inquiry at diamond 43 is NO, the process exits (end bubble 44). If the answer to this inquiry is YES, the process continues with a process step (block 45) of creating an audit filename. The process then appends the audit file to tape (block 46) and returns to diamond 43. The steps for appending an audit file to tape are further described in FIGS. 5A and 5B.

With reference to FIGS. 5A and 5B, there is seen a flowchart that illustrates the steps for appending an audit file to a Tapeset. This process is used for two purposes. First, the process is used for appending audit files when creating an initial Tapeset (refer to FIG. 4). Second, the process is used to append an audit file, or group of audit files, to an already existing Tapeset. The process begins with start bubble 50 and continues with a process step (block 51) of constructing a tape volume marker filename. The process uses the database name and Tapeset number when constructing the tape volume marker filename. Next, the process opens or creates the tape volume marker file by calling standard operating system functions (52). The process continues with a process step (block 53) of determining the name of the disk directory file. The process determines the name of the disk directory file by accessing an Associated Filename attribute on the tape volume. The Associated Filename attribute contains the exact location (i.e. title) of the disk file containing all the information about all the tape volumes comprising the Tapeset in question and also all the information about all the audit

awk/appl/558L.doc

files contained in the Tapeset. All volumes of the Tapeset inherit this attribute. Among the important pieces of data stored in the disk file are the first and last audit file numbers in the Tapeset and information about all the volumes in the Tapeset. After determining the name of the disk directory file, step block 53, the process continues with an inquiry as to whether or not the audit file number directly preceding the audit file number in question is found within the Tapeset (diamond 54). Because all audit files are sequential, the process makes this determination by checking if the preceding audit file number is less than the first audit file number within the Tapeset, or greater than the last audit file number within the Tapeset. If the answer to the inquiry posed by diamond 54 is no, the process exits (end bubble 55). If the answer to this inquiry is yes, the process continues by determining which tape volume within the Tapeset contains the preceding audit file number (block 56). Since the operating system could assign to the COPYAUDIT task any one of the tape volumes belonging to the Tapeset (the multi-reel case), it is imperative that the disk file contain sufficient information to find the other tape volumes in case a volume belonging to the Tapeset but not containing the required audit file is assigned to the program by the operating system. Each Volume Record entry in the disk directory file contains the first and last audit file numbers on that volume and the list of all audit files present on that volume. Thus, a simple scan of the disk file through the Volume Record entries reveals which particular tape reel contains the audit file in question. The Volume Record entry containing the audit file in question is examined

and the physical attributes (serial number, cycle and version) of the volume extracted. In addition, the Audit File Record entry is examined for the required attributes, namely, the 'starting' and 'ending' positions
5 of the audit file, extracted.

With reference to FIG. 5B, the process continues with an inquiry as to whether or not the tape volume containing the preceding audit file number is loaded (diamond 57). If the answer to this inquiry is
10 NO, the process closes the logical tape for the loaded tape volume (block 58), displays a message to load the needed tape volume name with the required attributes (serial number cycle, version) (block 59), and returns to decision diamond 57.

If the answer to the inquiry at diamond 57 YES,
15 the process continues with a process step (block 60), of using the fast locate capability of the tape drive to position the tape drive from the beginning of the tape volume to the end position of the preceding audit file
20 within the tape volume. The process then closes the tape file (block 61) and appends the audit file to tape (block 62). Next, the process updates the disk directory file (block 63). The steps involved with updating the disk directory file are further described in FIG. 6. After
25 updating the disk directory file, the process exits (end bubble 64).

Referring now to FIG. 6, the steps for updating the disk directory file are shown. The process begins with start bubble 70 and continues with a process step
30 (block 71) of creating an audit record entry in the disk directory file. The process then obtains a starting position of the audit file based on the audit file number

(block 72). Next, the process writes the starting position of the audit file to the audit record entry (block 73). Then at step 73a, there occurs the actual process of backing up the audit file to tape. The process then obtains the ending position of the audit file (block 74) and writes the ending position to the audit record entry (block 75). Before writing any data to the tape, the program creates an Audit Record entry in the disk file directory at the end of the current Audit record entries. It invokes the operating system directive, READPOSITION_DIRECT to obtain the "starting" position of the <audit file number being appended> and records it in the Audit Record entry. When the program finishes appending the audit file, it invokes the same operating system directive to obtain the "ending" position of the <audit file number being appended> and records it in the Audit Record entry. It updates the Directory Record entry and the Volume Record entry for the first and last audit file numbers. The process then exits (end bubble 76).

With reference to FIGS. 7A and 7B, the routine for retrieving an audit file from a Tapeset is shown. The process begins with start bubble 80 followed by a process step (block 81) of constructing a tape volume marker filename. For example, this file name might be designated as: <Database Name>/TAPESET<n>. The process uses the database name and Tapeset number (an integer) when constructing the tape volume marker filename. Next, the process opens the tape volume marker file by calling standard operating system functions (block 82), (i.e., File Open). The process then determines the name of the disk directory file using the Associated Filename

awk/appl/558L.doc

attribute. As discussed above, the Associated Filename attribute contains the exact location (i.e. title) of the disk file containing all the information about all the tape volumes comprising the Tapeset in question and also all the information about all the audit files contained in the Tapeset. The process then continues with an inquiry as to whether or not the audit file number in question is found within the Tapeset (diamond 84). Because all audit files are sequential, the process makes this determination by checking if the audit file number is less than the first audit file number within the Tapeset, or greater than the last audit file number within the Tapeset. If the answer to the inquiry posed by decision diamond 84 is NO, the process exits (end bubble 85). If the answer to this inquiry is YES, the process determines which tape volume contains the audit file number (block 86).

Since the operating system could assign to the COPYAUDIT utility program any one of the tape volumes belonging to the Tapeset (the multi-reel case), it is imperative that the disk file contain sufficient information to find the other tape volumes in case a (volume belonging to the Tapeset but not containing the required audit file) is assigned to the COPYAUDIT utility program by the operating system. Each Volume Record entry in the disk directory file contains the first and last audit file numbers on that volume and the list of all audit files present on that volume. Thus, a simple scan of the disk file through the Volume Record entries reveals which particular tape reel contains the audit file in question. The Volume Record entry containing the audit file in question is examined and the physical

attributes (serial number, cycle and version) of the volume extracted. In addition the Audit File Record entry is examined for the required attributes involving namely, the 'starting' and 'ending' positions of the audit file, extracted.

With reference to FIG. 7B, the process continues with an inquiry as to whether or not the needed tape volume is currently loaded (diamond 87). If the answer to this inquiry is NO, the process closes the logical tape for the current tape volume (block 88), displays a message to load the tape volume (with the required identifying attributes - serial number, cycle and version), (block 89) and returns to decision diamond 87.

If the answer to the inquiry at diamond 87 is YES, the process uses the fast-locate capabilities of the tape drive to move from the beginning of the tape volume to the ending position of the desired audit file minus 1 on the tape volume (block 90). Next, the process closes the tape file (block 91) and opens the desired audit file on the tape volume (block 92). Opening the audit file functions to create a logical association between the program and the physical file on tape. The process then exits (end bubble 93) after retrieving the desired audit files.

The methods and apparatus of the present invention, or certain aspects or portions thereof, may take the form of program code (i.e., instructions) embodied in tangible media, such as floppy diskettes, CD-ROMS, hard drives, or any other machine-readable storage medium, wherein, when the program code is loaded into and executed by a machine, such as a computer, the machine

becomes an apparatus for practicing the invention. The methods and apparatus of the present invention may also be embodied in the form of program code that is transmitted over some transmission medium, such as over electrical wiring or cabling, through fiber optics, or via any other form of transmission, wherein, when the program code is received and loaded into and executed by a machine, such as a computer, the machine becomes an apparatus for practicing the invention. When implemented on a general-purpose processor, the program code combines with the processor to provide a unique apparatus that operates analogously to specific logic circuits.

Although the invention has been described with reference to a specific embodiment, this description is not meant to be construed in a limiting sense. Various modifications of the disclosed embodiment as well as alternative embodiments of the invention will become apparent to one skilled in the art upon reference to the description of the invention. It is therefore contemplated that the appended claims will cover any such modifications or embodiments that fall within the true scope of the invention.